

DOCUMENT RESUME

ED 309 191

TM 013 662

AUTHOR Boekkooi-Timminga, Ellen
TITLE Algorithms for the Construction of Parallel Tests by Zero-One Programming. Project Psychometric Aspects of Item Banking No. 7. Research Report 86-7.
INSTITUTION Twente Univ., Enschede (Netherlands). Dept. of Education.
SPONS AGENCY Netherlands Inst. for the Advancement of Pure Research, The Hague.
PUB DATE 86
NOTE 35p.
AVAILABLE FROM Mediatheek, Faculteit Toegepaste Onderwijskunde, Universiteit Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands.
PUB TYPE Reports - Research/Technical (143)
EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS *Algorithms; *Computer Assisted Testing; Foreign Countries; Item Banks; *Latent Trait Theory; *Test Construction; Test Items
IDENTIFIERS Assignment Models; *Parallel Test Forms; *Zero One Programing

ABSTRACT

Nine methods for automated test construction are described. All are based on the concepts of information from item response theory. Two general kinds of methods for the construction of parallel tests are presented: (1) sequential test design; and (2) simultaneous test design. Sequential design implies that the tests are constructed one after the other; simultaneous test design considers the construction of all forms at the same time. The first two sequential methods (Methods 1 and 2) discussed do not assume any manipulation on the collection of items. One is based on constructing two tests following the same test item specifications, and the other assigns items in the second test to items in the first test. Four additional sequential methods (Methods 3 through 6) assume that the collection of items is divided into two comparable parts. These methods differ from the first two only in the specific groups of items from which the desired tests are determined. Three methods (Methods 7 through 9) are also presented from simultaneous test construction. Method 8 is based on the method of E. Boekkooi-Timminga (1986). The other two methods start from the principle of assignment. All of the methods are based on the use of the optimization technique zero-one programming. Illustrations of each of the nine methods indicate that the first two methods suggested may not be suitable for constructing parallel tests and that Method 8 most warrants further investigation. Three tables present test design methods and data from test selection. (SLD)

* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *

ED309191

Algorithms for the Construction of Parallel Tests by Zero-One Programming

Research
Report

86-7

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION
CENTER (ERIC)

- ☒ This document has been reproduced as received from the person or organization originating it.
- ☐ Minor changes have been made to improve reproduction quality.
- Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

"PERMISSION TO REPRODUCE THIS
MATERIAL HAS BEEN GRANTED BY

J. NELISSEN

TO THE EDUCATIONAL RESOURCES
INFORMATION CENTER (ERIC)."

E. Boekkooi-Timminga

Department of Education

Division of
Educational Measurement
and Data Analysis

University
of
Twente

TM013662

Project Psychometric Aspects of Item Banking No. 7

Colophon

Typing : L. Padberg
Cover design : M. Driessen, AV-section, University of Twente
Printed by : Central Reproduction Department, University of Twente

Algorithms for the Construction
of Parallel Tests by
Zero-One Programming

Ellen Boekkooi-Timminga

Abstract

Two kinds of methods for the construction of parallel tests are introduced, both based on the concept of information from item response theory. For all methods distinguished a test design model is described in terms of zero-one programming. The computational complexity of zero-one programming problems is discussed. Further an empirical example of each method is given. The paper ends with some concluding remarks and recommendations for further research.

Introduction

In classical test theory parallel tests play a prominent part, because most reliability coefficient estimation methods assume the existence of parallel tests. Some examples are internal-consistency methods such as those based on the Spearman-Brown formula, the Kuder-Richardson method, and the method of matched random subsets. Despite this central role of parallel tests hardly any algorithms for constructing such tests have been developed. One exception is the matched random subsets method proposed by Gulliksen (1950), which recently has been further algorithmized into an automated method by van der Linden and Boekkooi-Timminga (1968b). In item response theory very little attention is paid to parallel tests. This can be explained from the fact that no assumption of parallel measurements has to be made when item response models are used. In educational settings, however, there will always be a demand for parallel tests.

In this paper nine methods for automated parallel test construction are described all based on the concept of information from item response theory. The test items are selected from large collections of items using the optimization technique zero-one programming from operations research (e.g. Salkin, 1975; Taha, 1975; Wagner, 1972). Some previous research on algorithms for automated test design using information functions and zero-one programming, has already been carried out by Boekkooi-Timminga

(1986), Theunissen (1985), Theunissen and Verstralen (1986), Timminga (1985) and van der Linden and Boekkooi-Timminga (1986a). For an exact definition of the concept of information the reader is referred to Birnbaum (1968, chapter 17), or Lindgren (1976, sect. 4.5.4). For the three-parameter model the information function of a dichotomous item is given by

$$(1) \quad I_i(\theta) = \frac{a_i^2(1-c_i)}{(c_i + \exp(a_i(\theta - b_i)))(1 + \exp(-a_i(\theta - b_i)))^2},$$

where a_i , b_i and c_i indicate the discrimination, difficulty and guessing level of item i and θ the subject's ability level. The test information function $I(\theta)$ is simply the sum of the item information functions of the test items, provided maximum-likelihood estimators are used for the subjects' abilities.

In this paper, as well as in the ones referred to above, it is assumed that collections of calibrated items all fitting the same one-dimensional model are available. Furthermore, all methods for test construction described expect the test constructor to specify at some ability levels a target value for the information function of the test(s) to be constructed.

Two general kinds of methods for the construction of parallel tests are presented: sequential and simultaneous test design. In either, tests are considered to be parallel when their information functions are the same (Samejima, 1977). Sequential test design implies that the tests are constructed one after another. Simultaneous test design considers the construction of all tests at

the same time. First, six sequential and three simultaneous methods are described; then, after a discussion of the computational complexity of zero-one programming problems, for each of them an empirical example is given. A discussion concludes the paper.

Nine Methods for the Construction of Parallel Tests

Six sequential and three simultaneous methods for parallel test design are distinguished. Table 1 gives a summary of all nine methods. In this section, first, a list of symbols used is given; then, the two groups of methods are described. For convenience only the construction of two parallel tests is considered. Each method, though, can be applied also when more than two parallel tests have to be constructed; in the discussion of this paper this is described briefly.

Notation

Constants

- i : items in the item bank ($i = 1, \dots, N$)
- n : number of items to be selected
- t : tests to be constructed ($t = 1, \dots, T$)
- k : points at the ability scale considered ($k = 1, \dots, K$)
- I_{ik} : information of item i at ability level k
- I_k : target test information at ability level k
- c_{ij} : difference between item i and j

Variables

x_i	: decision variable	$\begin{cases} 1 & \text{item } i \text{ is selected for the test} \\ 0 & \text{otherwise} \end{cases}$
x_{it}	: decision variable	$\begin{cases} 1 & \text{item } i \text{ is selected for test } t \\ 0 & \text{otherwise} \end{cases}$
x_{ij}	: decision variable	$\begin{cases} 1 & \text{item } j \text{ is chosen as parallel item of} \\ & \text{item } i \\ 0 & \text{otherwise} \end{cases}$
y	: help variable, indicating a maximum distance between test information functions	

Insert Table 1 about here

Sequential Test Design

Two groups of sequential test design methods are described (see Table 1). Group B assumes that the collection of items from which the tests have to be constructed is divided into two comparable parts: for instance, in terms of the distribution of difficulty or discrimination indices over both parts. No ordering of the items within these parts is assumed. Group A does not assume any manipulation on the collection of items.

Group A consists of two methods. Both methods start from the

same principle: First, one test is constructed; then, the second test is selected from the whole collection of items excluding the items already chosen for the first test. Method 1 is based on constructing two tests both following the same test specifications. In the case that the maximum distance y between the target and actual information function at the k ability levels considered is minimized, the test construction model for the determination of the first test is formulated as

(2) minimize y

subject to

$$(3) \quad \sum_{i=1}^N x_i I_{ik} - y \leq I_k \quad k = 1, \dots, K$$

$$(4) \quad \sum_{i=1}^N x_i I_{ik} \geq I_k \quad k = 1, \dots, K$$

$$(5) \quad y \geq 0$$

$$(6) \quad x_i \in \{0, 1\} \quad i = 1, \dots, N,$$

where (2) in combination with (3) indicates that the maximum distance is minimized. In (4) the target values for the test information function are defined to be at least I_k at the k ability levels. The limits on the values of y and the x_i 's are set by (5) and (6). In zero-one programming expression (2) is called the

objective function and (3) to (6) are called the constraints. The model for constructing the second test is the same as the one above, except for some constraints that have to be added: excluding the items i selected for the first test

$$(7) \quad x_i = 0$$

and if it is desired that both tests should contain an equal number of items

$$(8) \quad \sum_{i=1}^N x_i = n,$$

where n is the number of items in test one.

Method 2 (Group A) differs from Method 1 in the determination of the second test. Those items j are selected for the second test that correspond best to the items i in the first test. Constructing the second test can, thus, be seen as an assignment problem, in which the items for the second test are assigned to those in the first test. The test design model for the second test then becomes

$$(9) \quad \text{minimize} \quad \sum_{i=1}^n \sum_{j=1}^N c_{ij} x_{ij}$$

subject to

$$(10) \quad \sum_{i=1}^n x_{ij} \leq 1 \quad j = 1, \dots, N$$

- (11) $\sum_{j=1}^N x_{ij} = 1$ $i = 1, \dots, n$
- (12) $\sum_{i=1}^n x_{ij} = 0$ for all items j
that belong to
test one
- (13) $x_{ij} \in \{0, 1\}$ $i = 1, \dots, n$
 $j = 1, \dots, N$
- (14) $\sum_{i=1}^n \sum_{j=1}^N I_{jk} x_{ij} \geq I_k$ $k = 1, \dots, K.$

The items selected for the first test are $i = 1, \dots, n$. The difference between the items i and j is indicated by c_{ij} . All kinds of difference measures are allowed, for instance, the sum of the absolute differences in item information at the ability levels considered, or the difference in item difficulty. As an objective function (9) minimizes the sum of the differences between the items in the first test and the corresponding items in the second test. The constraints (10) and (11) indicate that for each item in the first test exactly one corresponding item in the second test should be selected. Those items that have already been chosen for test one are excluded from test two through (12). By adding constraint (14), which is optional, it is possible to specify the target test information function of the second test.

The second group of methods (Group B) in sequential parallel test design, consists of four methods. As already mentioned, these methods assume that the collection of items is divided into two comparable parts. The four methods in Group B differ only from

Methods 1 and 2 in the specific group of items from which the desired tests are determined. Thus, the same test design models are appropriate provided they are adapted to the group of items from which the selection should take place. Like Method 1 the same test specification is used in Methods 3 and 5. In Methods 4 and 6 the tests are selected using the assignment principle as in Method 2. Furthermore, in Methods 3 and 4 both tests are constructed each from different parts of the collection of items, whereas, in Methods 5 and 6 the first test is selected from one part and the second test from the whole collection of items excluding the items in the first test.

Simultaneous Test Design

In simultaneous test design parallel tests are constructed at the same time. Three methods for simultaneous parallel test construction are distinguished, all summarized in Table 1. The first method to be described is based on the same idea of simultaneous test design as discussed in Boekkooi-Timminga (1986). The other two methods are both starting from the principle of assignment.

Method 7 consists of defining for each of the parallel tests to be constructed the same target test information function. Further, it is indicated that the items belonging to the collection of items may be selected only for one of the tests. A possible test construction model for the case of two parallel tests is

(15) minimize y

subject to

$$(16) \quad \sum_{i=1}^N x_{i1} I_{ik} - y - \sum_{i=1}^N x_{i2} I_{ik} < 0 \quad k = 1, \dots, K$$

$$(17) \quad -\sum_{i=1}^N x_{i1} I_{ik} - y + \sum_{i=1}^N x_{i2} I_{ik} < 0 \quad k = 1, \dots, K$$

$$(18) \quad \sum_{i=1}^N x_{it} I_{ik} > I_k \quad k = 1, \dots, K$$

$$t = 1, 2$$

$$(19) \quad \sum_{t=1}^2 x_{it} < 1 \quad i = 1, \dots, N$$

$$(20) \quad y > 0$$

$$(21) \quad x_{it} \in \{0, 1\} \quad i = 1, \dots, N$$

$$t = 1, 2$$

$$(22) \quad \sum_{i=1}^N x_{i1} - \sum_{i=1}^N x_{i2} = 0,$$

where (15), (16) and (17) indicate that the maximum absolute distance y between the actual test information function values for both parallel tests is minimized. Constraint (18) defines the target values for the test information functions, and (19) indicates that the items belonging to the collection of items may be selected only once. If it is required that the parallel tests should contain an equal number of items, (22) should be added. Using this method, other objective functions can also be applied.

Some of them are exclusively to be used for test construction models belonging to Method 7. These are expressions considering the actual test information functions of both tests to be constructed, as for instance, the objective function in (15) - (22).

The second method for simultaneous test design is Method 8 which is based on the same principle of assignment as used in Methods 2, 4 and 6. The difference is that none of the items in one test for which a corresponding item has to be found in the other is known beforehand. A model formulation for this method is

$$(23) \text{ minimize } \sum_{i=1}^N \sum_{j=1}^N c_{ij} x_{ij}$$

subject to

$$(24) \quad \sum_{i=1}^N x_{ij} + \sum_{i=1}^N x_{ji} < 1 \quad j = 1, \dots, N$$

$$(25) \quad \sum_{i=1}^N \sum_{j=1}^N I_{ik} x_{ij} > I_k \quad k = 1, \dots, K$$

$$(26) \quad \sum_{i=1}^N \sum_{j=1}^N I_{jk} x_{ij} > I_k \quad k = 1, \dots, K$$

$$(27) \quad x_{ij} \in \{0, 1\} \quad \begin{array}{l} i = 1, \dots, N \\ j = 1, \dots, N, \end{array}$$

where c_{ij} is large whenever $i = j$. By this test construction model, two parallel tests can be constructed for which the sum of the differences c_{ij} between the corresponding items of the tests is

minimal. For both tests the same target test information function is specified in (25) and (26). Constraint (24) indicates that each item may be selected only once. The decision variables x_{ij} should be interpreted here as follows:

$$x_{ij} = \begin{cases} 1 & \text{item } i \text{ and } j \text{ are chosen as corresponding} \\ & \text{items, where item } i \text{ and } j \text{ are selected for} \\ & \text{the first and second test, respectively} \\ 0 & \text{otherwise} \end{cases}$$

Method 9, just as Methods 3 to 6, is based on dividing the collection of items into two comparable parts. From each part of the collection of items a test is selected using the same assignment principle as in Methods 2, 4, 6 and 8. It is assumed that both parts contain an equal number of items. Part one and two consist of the items 1, 2, ..., $N/2$ and $N/2 + 1$, ..., N , respectively. The test construction model for this method is

$$(28) \text{ minimize } \sum_{i=1}^{N/2} \sum_{j=N/2+1}^N c_{ij} x_{ij}$$

subject to

$$(29) \quad \sum_{i=1}^{N/2} x_{ij} < 1 \quad j = N/2 + 1, \dots, N$$

$$(30) \quad \sum_{j=N/2+1}^N x_{ij} < 1 \quad i = 1, \dots, N/2$$

$$(31) \quad \sum_{i=1}^{N/2} \sum_{j=N/2+1}^N I_{ik} x_{ij} > I_k \quad k = 1, \dots, K$$

$$(32) \quad \sum_{i=1}^{N/2} \sum_{j=N/2+1}^N I_{jk} x_{ij} > I_k \quad k = 1, \dots, K$$

$$(33) \quad x_{ij} \in \{0, 1\} \quad \begin{array}{l} i = 1, \dots, N/2 \\ j = N/2 + 1, \dots, N \end{array}$$

The sum of the differences between the corresponding items in the two parallel tests is minimized in (28). Items are prevented from being selected more than once by constraints (29) and (30). The target test information functions, which are the same for both parallel tests to be constructed, are defined by (31) and (32).

Computational Complexity

The methods for parallel test construction described above all are based on the use of the optimization technique zero-one programming. Zero-one programming can be seen as a special case of discrete or integer programming, which itself is a branch of linear programming. The difference between these techniques is the range of values the decision variables are allowed to take. Linear programming allows the decision variables to take any nonnegative real value. Discrete and zero-one programming only permit nonnegative integer values and the values zero and one to occur, respectively.

Linear programming problems are considered to be P-complete,

which means that fast algorithms bounded by a polynomial function of the problem size exist. An example is the well-known simplex algorithm. Discrete programming problems have proved to be much harder to solve, and it is commonly conjectured that for the general discrete programming problem no fast algorithm exists as yet (Lenstra & Rinnooy Kan, 1979). Only for some specially structured problems highly efficient algorithms have been developed. The test construction problems described in this paper, unfortunately, do not have such special structures. They can be looked at as combinatorial problems that are NP-complete, which indicates that the amount of time needed to solve these problems is not bounded by a polynomial function but will be larger.

Because most discrete programming problems meet this problem of computational complexity, many research has been carried out in this area. A comprehensive overview of the literature describing this research is given in O'hEigeartaigh, Lenstra and Rinnooy Kan (1985). Since it is generally accepted that no fast algorithms will exist for NP-complete problems, much research aims at approximations. Considering the large amount of research adressing this problem, and the number of results already obtained during the last decade, it is expected that fast approximative algorithms will be found before long.

Some Empirical Examples

As an illustration for each of the nine methods described an

example is given. The algorithm used for solving the test construction problems considered was a branch-and-bound algorithm developed by Land and Doig (1960) implemented on a DEC-2060 computer. Since the meaning of these examples was to test the design models on the computer and explore their behavior, the domain of items used to construct the tests from consisted of only 14 items. Applications to more realistic domains of items have to wait solutions to the computational complexity of zero-one programming problems discussed in the previous section. The item parameters and the item information values at the ability levels $\theta = -1, 0, 1$ of the items are given in Table 2 .

Insert Table 2 about here

For all items it was assumed that there was no guessing; the other item parameters were chosen at random. Table 2 shows that the item difficulties b_i varied from -1.708 to 1.828, and the item discrimination indices a_i from 0.549 to 1.384.

In the cases of Methods 1, 3, 5, and 7 two examples were described: one without (tests 1-1a, 3-3a, 5-5a, 7a-7a') and the other with (tests 1-1b, 3-3b, 5-5b, 7b-7b') the constraint that the both tests to be constructed should contain the same number of items. For Methods 2, 4, and 6 also two examples were given; now one without (tests 2-2a, 4-4a, 6-6a) and the other with (tests 2-2b, 4-4b, 6-6b) an indication of the same target test information

function as the first test when the second test was constructed.

The target test information functions were the same for all methods. At the three ability levels mentioned the target values were $I_k > 0.4$. When Methods 2, 4, 6, and 8 were applied, as a measure for the differences c_{ij} between items i and j the sum of the squared absolute differences in item information over the three ability levels was considered. In the examples using Methods 3 to 6, the collection of items was divided into two comparable parts by first ordering the items on difficulty and then putting all items with even and odd numbers in different parts. Part one of the collection of items then consisted of the items 2, 5, 6, 7, 9, 11, 14 and part two of the items 1, 3, 4, 8, 10, 12, 13.

The results of the examples are summarized in Table 3. For each test constructed six results are given: (1) the items selected, (2) the number of items n selected, (3) the test information function values at the three ability levels, (4) the optimum value z of the objective function minimized, (5) the maximum distance y between the actual and target test information function values, and (6) the maximum distance y^* between the actual test information function values of both parallel tests constructed. In the cases in which the tests were selected by assignment (Methods 2, 4, 6, 8, 9) it can easily be seen in Table 3 which items were chosen to be parallel by the method. For instance, Method 9 produced the item pairs (6-13), (7-4), and (14-12).

Insert Table 3 about here

Care should be taken in comparing the results in Table 3 because the objective functions differed for some methods. The maximum distance between the target and actual test information function values was minimized when tests 1, 1a, 1b, 2, 3, 3a, 3b, 4, 5, 5a, 5b, 6 were determined. The sum of the differences c_{ij} between the items in both parallel tests was minimized for tests 2a, 2b, 4a, 4b, 6a, 6b, 8, 9. And, finally, the maximum distance between the actual test information function values of both parallel tests was minimized when Method 7 was considered. Despite the fact that the results are also strongly influenced by the specific target test information function desired and the collection of items used, the following conclusions can be drawn.

As mentioned earlier, tests are defined to be parallel if their test information functions are the same. The values y^* , therefore, should be minimal. In comparing the sequential and simultaneous test design methods it is seen that the two methods with the smallest y^* values were both simultaneous methods, namely Methods 7 and 8. Method 7 explicitly minimized the maximum distance between both actual test information functions thus obtaining the smallest value for y^* . The number of items selected and the distances y , however, were less ideal. Therefore, if a minimum number of items as well as low y values are desired, Method 8 is concluded to be the best method for parallel test construction under the

circumstances of these examples.

Looking at the sequential test design methods, it is clearly seen that Methods 1 and 2 gave the worst results for y^* and y values and the number of items selected. By partitioning the collection of items, considerable improvement was made; especially Methods 3, 5 and 6 resulted in relatively small y^* values. Conclusions on which of these methods is best should better not be made, because the results were strongly influenced by the small collection of items used, the fact that the first test was constructed from part one and not part two of the collection of items (compare the results of Methods 3, 5, 6 with those of Method 8), and the objective function used.

It is concluded that Methods 1 and 2 are not suitable for constructing parallel tests. Method 8 seems to be most promising. However, before general conclusions can be drawn, further research is needed on the behavior of this as well as the other six methods.

Discussion

The purpose of this paper was to introduce some methods for constructing parallel tests from collections of items using information functions. The actual item selection was based on the use of the optimization technique zero-one programming. The nine methods presented were split into two groups: sequential and simultaneous test design methods. For each of the methods a zero-

one programming model for the construction of two parallel tests was described and an example was given.

As already mentioned these methods can also be applied when more than two parallel tests should be constructed. How this should be done is mentioned briefly. As already described, when Methods 1 and 2 are considered, the process of test construction is repeated until the number of tests to be selected is reached. When Methods 3 to 6 are used the collection of items should be divided into as many comparable parts as tests to be constructed; the item selection process is then continued in the way described earlier. In Method 7 index t should be adapted to the number of tests to be constructed. Constraints like (16) and (17) should be added to account for all possible distances y between the actual test information functions of the tests to be constructed. Finally, when it is desired that all tests should contain an equal number of items, constraint (22) should be adapted. When Methods 8 and 9 are to be used with more than two tests, the variables x_{ij} and c_{ij} should be changed as well. In the case of three parallel tests they become x_{ijm} and c_{ijm} indicating that the items i , j , and m are chosen as parallel items with c_{ijm} is a measure for the difference between the items i , j , m . Furthermore, in Method 8, (23) and (24) should be replaced by

$$(34) \quad \text{minimize} \quad \sum_{i=1}^N \sum_{j=1}^N \sum_{m=1}^N c_{ijm} x_{ijm},$$

and

$$(35) \quad \sum_{i=1}^N \sum_{j=1}^N x_{ijm} + \sum_{i=1}^N \sum_{j=1}^N x_{imj} + \sum_{i=1}^N \sum_{j=1}^N x_{mij} < 1$$

$$m = 1, \dots, N,$$

The constraints (25) and (26) should be adapted and the following one like them should be added

$$(36) \quad \sum_{i=1}^N \sum_{j=1}^N \sum_{m=1}^N I_{mk} x_{ijm} > I_k \quad k = 1, \dots, K.$$

Using Method 9 the collection of items should be divided into as many parts as tests to be constructed. In the case of three tests the parts consist of the items $1, \dots, N/3$, $N/3 + 1, \dots, 2N/3$, and $2N/3 + 1, \dots, N$. Objective function (28), then, should take the sum over all these items, were the variables x_{ij} and c_{ij} are transformed into x_{ijm} and c_{ijm} . Constraints (29), (30), (31) and (32) should be adapted, and constraint (37) like (29) and (30) and (38) like (31) and (32) should be added

$$(37) \quad \sum_{m=2N/3+1}^N x_{ijm} < 1 \quad i = 1, \dots, N/3,$$

and

$$(38) \quad \sum_{i=1}^{N/3} \sum_{j=N/3+1}^{2N/3} \sum_{m=2N/3+1}^N I_{mk} x_{ijm} > I_k \quad k = 1, \dots, K$$

Instead of the objective functions used in the test design models described here, several other functions can be used. Further, it is possible to add all kinds of practical constraints, e.g., with respect to the number of items to be selected. For a report on some of the possibilities the reader is referred to van der Linden and Boekkooi-Timminga (1986a), and to Boekkooi-Timminga (1986) for simultaneous test design according to Method 7.

In this paper tests are defined to be parallel only with respect to their psychometric properties. By introducing indicator variables, indicating for each item if a certain subject matter is covered or not, it is also possible to consider tests to be parallel with respect to their content. Constraints can be put on the ratio in which items from certain topics should be selected for the tests to be constructed. Let S be the number of topics, and p_{ik} the indicator variable for item i and topic s , which is one if item i covers topic s and zero otherwise. Constraint (39), for instance to be used with Method 7, indicates that the ratio $a_1, a_2, \dots, a_S$ in which items from certain topics s should be selected must be the same for all tests.

$$(39) \quad a_1 \sum_{i=1}^N p_{i1} x_{it} = a_2 \sum_{i=1}^N p_{i2} x_{it} = \dots = a_S \sum_{i=1}^N p_{iS} x_{it} \quad t = 1, \dots, T$$

The purpose of the examples described in this paper was not to conclude which method is best for constructing parallel tests. They were given to show the specific behavior of each method. A first careful conclusion, which needs more research, recommends method 8.

Simulation studies are needed on how the methods behave when different objective functions are used. In these studies large collections of items should be used that vary systematically, because variations in the distributions of the item parameter indices also influence the selection process as is shown by Timminga (1985). Other research should aim at the problem of specifying a target test information function; a first step to solve this problem is given in Kelderman (1986). The algorithms used to solve the test construction problems also require extra research on how to speed them up or simplify them. And finally research on quick heuristics that give solutions close to optimal is needed; some research on this problem directly pointed at the problem of test construction has already been done by Boomsma (1986).

References

- Birnbaum, A. (1968). Some latent trait models and their use in inferring an examinee's ability. In F.M. Lord & M.R. Novick, Statistical theories of mental test scores. London: Addison-Wesley Publishing Company, Inc.
- Boekkooi-Timminga, E. (1986). Simultaneous test construction by zero-one programming. Methodika, 1.
- Boomsma, Y. (1986). Item selection by mathematical programming. Enschede, The Netherlands: Department of Applied Mathematics, University of Twente.
- Gulliksen, H. (1950). Theory of mental tests. New York: John Wiley & Sons, Inc.
- Kelderman, H. (1986). A procedure to assess information functions for the construction of tests measuring multiple traits. Enschede, The Netherlands: Department of Education, University of Twente.
- Land, A.H., & Doig, A. (1960). An automatic method of solving discrete programming problems. Econometrica, 28, 497-520.
- Lenstra, J.K., & Rinnooy Kan, A.H.G. (1979). Computational complexity of discrete optimization problems. In P.L. Hammer, E.L. Johnson, & B.H. Korte (Eds.), Discrete optimization I. New York: North-Holland Publishing Company.
- Lindgren, B.W. (1976). Statistical theory (3rd ed.). New York: MacMillan Publishing Co., Inc.

- Lord, F.M. (1980). Applications of item response theory to practical testing problems. Hillsdale, New Jersey: Lawrence Erlbaum Associates, Publishers.
- O'hEigeartaigh, M., Lenstra, J.K., & Rinnooy Kan, A.H.G. (Eds.). (1985). Combinatorial optimization: annotated bibliographies. New York: John Wiley & Sons.
- Samejima, F. (1977). A use of the information function in tailored testing. Applied Psychological Measurement, 1, 233-247.
- Salkin, H.M. (1975). Integer programming. London: Addison-Wesley Publishing Company.
- Taha, H.A. (1975). Integer programming. New York: Academic Press.
- Theunissen, T.J.J.M. (1985). Binary programming and test design. Psychometrika, 50, 411-420.
- Theunissen, T.J.J.M., & Verstralen, H.H.F.M. (1986). Algoritmen voor het samenstellen van toetsen Algorithms for test construction . In W.J. van der Linden (Ed.), Moderne methoden voor toetsgebruik- en constructie. Lisse: Swets & Zeitlinger
- Timminga, E. (1985). Geautomatiseerd toetsontwerp: Itemselectie m.b.v. binair programmeren Automated test design: Item selection by binary programming . Enschede, The Netherlands: Department of Education, University of Twente.
- van der Linden, W.J., & Boekkooi-Timminga, E. (1986a). Algorithmic test design with practical constraints. Enschede, The Netherlands: Department of Education, University of Twente.

van der Linden, W.J., & Boekkooi-Timminga, E. (1986b). A zero-one programming approach to Gulliksen's matched random subsets method. Enschede, The Netherlands: Department of Education, University of Twente.

Wagner, H.M. (1972). Principles of operations research: with applications to managerial decisions. London: Prentice-Hall International, Inc.

Table 1
Test Design Methods for Parallel Tests

SEQUENTIAL TEST DESIGN

A. No partition of the collections of items

1. twice the same test specifications
2. second test is determined by assigning the items to the items in the first test.

B. Partition of the collection of items in two parts

Selecting a test from each part

3. twice the same test specifications
4. second test is determined by assigning the items to the items in the first test.

Selecting the first test from part one and the second test from the whole collections of items excluding the items in the first test

5. twice the same test specifications
6. second test is determined by assigning the items to the items in the first test.

SIMULTANEOUS TEST DESIGN

7. The tests are selected at the same time from the whole collection of items both having the same target test information function.
8. Those tests are selected by assignment for which the sum of differences between the corresponding items is minimal, and that satisfy the target test information functions.
9. A partition of the collection of items in two parts is assumed: from each part a test is selected for which the sum of the differences between the corresponding items of the test is minimal by assignment.

Table 2
Item Parameters and Item Information Values

Item	b_i	a_i	$I_i(-1)$	$I_i(0)$	$I_i(1)$
1	0.576	0.695	0.091	0.116	0.118
2	-0.442	1.109	0.280	0.290	0.172
3	-0.824	0.823	0.168	0.151	0.101
4	0.254	0.609	0.080	0.092	0.088
5	0.419	1.213	0.189	0.345	0.326
6	-0.017	1.138	0.240	0.324	0.236
7	-0.245	0.549	0.072	0.075	0.067
8	1.828	1.171	0.047	0.129	0.273
9	1.109	0.892	0.091	0.157	0.198
10	-0.080	0.879	0.165	0.193	0.155
11	-1.708	1.384	0.380	0.151	0.043
12	0.016	0.909	0.168	0.207	0.170
13	-0.264	1.299	0.339	0.410	0.229
14	0.063	0.936	0.173	0.219	0.182

Table 3
Parallel Tests Selected

Method	Test	Items Selected	n	I(-1)	I(0)	I(1)	z	y	y*
1	1	3-4-7-9	4	0.411	0.475	0.454	0.075	0.075	
	1a	1-10-12	3	0.424	0.516	0.443	0.116	0.116	0.041
	1b	1-8-10-11	4	0.683	0.589	0.589	0.283	0.283	0.272
2	2	3-4-7-9	4	0.411	0.475	0.454	0.075	0.075	
	2a	2-1-12-11	4	0.919	0.764	0.503	0.418	0.364	0.508
	2b	2-1-12-11	4	0.919	0.764	0.503	0.418	0.364	0.508
3	3	2-7-9	3	0.443	0.522	0.437	0.122	0.122	
	3a	4-10-12	3	0.413	0.492	0.413	0.092	0.092	0.030
	3b	4-10-12	3	0.413	0.492	0.413	0.092	0.092	0.030
4	4	2-7-9	3	0.443	0.522	0.437	0.122	0.122	
	4a	10-3-1	3	0.424	0.460	0.374	0.496	0.060	0.063
	4b	13-3-1	3	0.598	0.677	0.448	.0496	0.277	0.155
5	5	2-7-9	3	0.443	0.522	0.437	0.122	0.122	
	5a	1-3-14	3	0.432	0.486	0.401	0.086	0.086	0.036
	5b	1-3-14	3	0.432	0.486	0.401	0.086	0.086	0.036
6	6	2-7-9	3	0.443	0.522	0.437	0.122	0.122	
	6a	14-3-1	3	0.432	0.486	0.401	0.402	0.086	0.036
	6b	14-3-1	3	0.432	0.486	0.401	0.402	0.086	0.036
7	7a	9-10-12-14	4	0.597	0.776	0.705		0.376	
	7a'	1-3-4-5-7	5	0.600	0.779	0.700	0.005	0.379	0.005
	7b	3-7-9-13-14	5	0.843	1.012	0.777		0.612	
	7b'	1-2-4-6-10	5	0.856	1.015	0.769	0.013	0.615	0.013
8	8	1-3-14	3	0.432	0.486	0.401		0.086	
	8'	4-10-12	3	0.413	0.492	0.413	0.0066	0.092	0.019
9	9	6-7-14	3	0.485	0.618	0.485		0.218	
	9'	13-4-12	3	0.587	0.709	0.487	0.0184	0.309	0.102

n: number of items selected

z: optimum objective function value

y: maximum distance between the actual and the target test information function values

y*: maximum distance between the actual test information function values of both parallel tests.

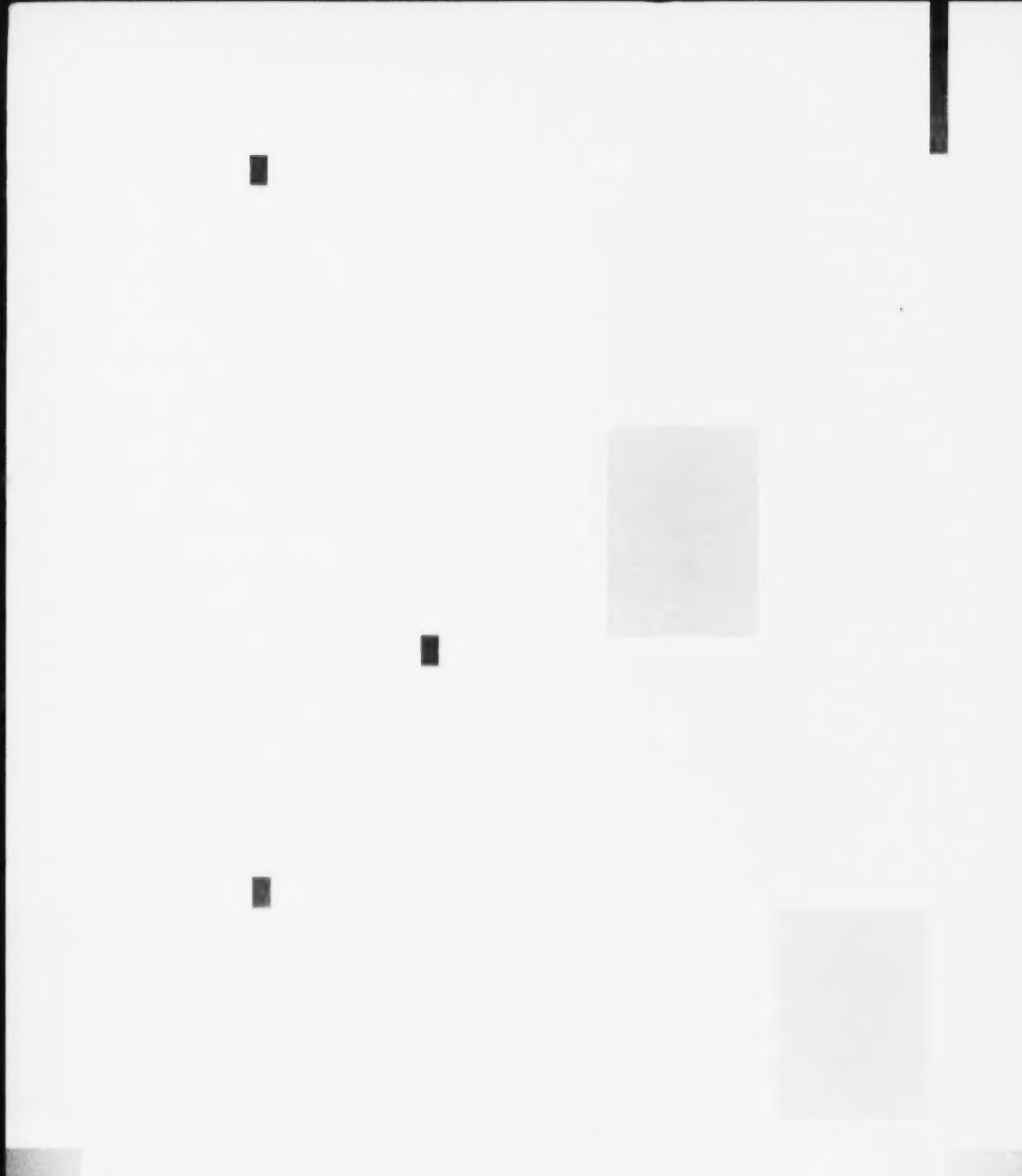
Author's Note

This research was supported by a grant from the Dutch Organization for Pure Research (Z.W.O.) through the Foundation for Psychological and Psychonomic Research in the Netherlands (PSYCHON).

Requests for reprints should be sent to Ellen Boekkooi-Timminga, Faculteit der Toegepaste Onderwijskunde, Universiteit Twente, Postbus 217, 7500 AE Enschede, The Netherlands.

Titles of Recent Research Reports

- RR-86-1 W.J. van der Linden, The use of test scores for classification decisions with threshold utility
- RR-86-2 H. Kelderman, Item bias detection using the loglinear Rasch model: Observed and unobserved subgroups
- RR-86-3 E. Boekkooi-Timminga, Simultaneous test construction by zero-one programming
- RR-86-4 W.J. van der Linden, & E. Boekkooi-Timminga, A zero-one programming approach to Gulliksen's random matched subtests method
- RR-86-5 E. van der Burg, J. de Leeuw, & R. Verdegaal, Homogeneity analysis with k sets of variables: An alternating least squares method with optimal scaling features
- RR-86-6 W.J. van der Linden, & T.J.H.M. Eggen, An empirical Bayes approach to item banking
- RR-86-7 E. Boekkooi-Timminga, Algorithms for the construction of parallel tests by zero-one programming
- RR-86-8 T.J.H.M. Eggen, & W.J. van der Linden, The use of models for paired comparisons with ties



A publication by
the Department of Education
of the University of Twente,
P.O. Box 217,
7500 AE Enschede,
The Netherlands

Department of Education